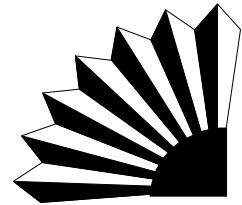


the Technical Broadcast



Published by the Department of Information Services

Winter/Spring 1996

COBOL/370 and LE/370, Article Four

 by Gary Duffield

Fourth in a series of eight articles on features of the COBOL/370 and LE/370 programming language, this article addresses REFERENCE MODIFICATION, another feature common to COBOL II and COBOL/370, and INTRINSIC FUNCTIONS, a feature unique to COBOL/370.

INTRINSIC (inherent, within) FUNCTIONS (logic) create mini-programs that are embedded within COBOL/370. These programs allow character and number handling, date and time processing, and statistical and mathematical calculations.

Since the functions are 'intrinsic,' instead of CALLing them, you simply refer to them with the FUNCTION keyword and pass some parameters within parentheses. Be sure to pass them the data they expect! If they abend, it's not always immediately obvious to you what happened.

There are some 42 of these intrinsic functions. Let's take a look at a sample of each type:

Character Handling

Imagine your program receives data containing last names, but in all capital letters. Ugly! You'd like to convert all but the first letter to lower case. Note: Last names requiring more than just the first letter capitalized would have to be handled differently.

This is greatly simplified in COBOL/370 by the use of the LOWER-CASE function and REFERENCE MODIFICATION.

EXAMPLE:

(Continued on page 2)

Inside...

It's a Disaster	6	DIS Partners with the State Treasurer	14
Making a Jump to Message Control Bank (MCB)	9	Why Restore an Archived Data Set just to Delete It?	17

COBOL/370 and LE/370, Article Four

(Continued from page 1)

```
05 LAST-NAME          PIC X(25).  
  
MOVE FUNCTION LOWER-CASE(LAST-NAME) (2:24) TO  
    LAST-NAME (2:24).
```

The LOWER-CASE function converts characters to lower case, but what restricts it from modifying the first character? REFERENCE MODIFICATION.

The MOVE statement by nature wants to move the entire LAST-NAME field. By definition, it would start in position 1 and grab a length of 25 characters. However, by adding those little parentheticals, we ‘modify’ its ‘reference’ to each field. We tell it to move starting in position 2 for a length of 24 and put the characters down in position 2 for a length of 24. It just so happens that in this example we do the move right back on top of the field, so the first character is still there in its untouched capital state.

Of course, some pessimistic person will say, “Yeah, and then on the day that the length of the LAST-NAME field changes, nobody remembers that parenthetical reference modifier down in the logic which then becomes invalid, and who knows what strange things will occur.”

Well, if you want to be that way, why don’t we use another function to remedy that? The LENGTH function:

```
MOVE FUNCTION LOWER-CASE(LAST-NAME)  
    (2:FUNCTION LENGTH(LASTNAME) - 1)  
    TO LAST-NAME  
    (2:FUNCTION LENGTH(LAST-NAME) - 1).
```

Okay, so putting another function right in the middle of a reference modifier is not very pretty, but at least you won’t ever have to worry if the field length changes!

Number Handling

Suppose you have character fields that contain edited dollar amounts -- you know, with commas, decimal point, and dollar sign. It sure would be useful to be able to extract the numeric part for math purposes, wouldn’t it? Welcome the ‘Numeric Value’ (NUMVAL) and ‘Numeric Value Currency’ (NUMVAL-C) functions!

(Continued on page 3)

COBOL/370 and LE/370, Article Four

(Continued from page 2)

Consider the following data-names in a program:

05 UNIT-PRICE	PIC X(10).
05 SHIP-HANDLE	PIC X(6).
05 TOTAL-COST	PIC 9(5)V99 PACKED-DECIMAL.

The UNIT-PRICE has a value of '\$1,023.99' and SHIP-HANDLE a value of '+8.75'. Normally, those currency symbols and commas meant you had to do quite a bit of work before you could do any math. Now you just invoke the NUMVAL functions to pick out the numbers for you and you can add up the result (1032.74) in TOTAL-COST:

```
COMPUTE TOTAL-COST =  
    FUNCTION NUMVAL-C(UNIT-PRICE) +  
    FUNCTION NUMVAL(SHIP-HANDLE).
```

By the way, did you notice the new data type phrase PACKED-DECIMAL? Doesn't that make a lot more sense than COMP-3? It means the same thing.

Date and Time Handling

What will be the date 42 days from today? Want to write the logic to supply the answer? Nah. Use INTRINSIC FUNCTIONS instead!:

05 ANSWER	PIC X(8).
05 ANSWER-INTEG	REDEFINES ANSWER
	PIC 9(8).
MOVE FUNCTION CURRENT-DATE TO ANSWER	
COMPUTE ANSWER-INTEG =	
FUNCTION DATE-OF-INTEG	
(FUNCTION INTEG-OF-DATE(ANSWER-INTEG) + 42)	

(Continued on page 4)

COBOL/370 and LE/370, Article Four

(Continued from page 3)

The first MOVE uses the CURRENT-DATE function to get us today's date in the form YYYYMMDD.

Working the COMPUTE from the innermost parenthesis out, we use the function INTEGER-OF-DATE to convert today's date into an integer representing the number of days since January 1, 1601. To that we add 42. Then we use the function DATE-OF-INTEGER to turn it back into the YYYYMMDD format.

Simple, isn't it? To put it in perspective, imagine writing all the logic yourself. Now, to ask again ... simple, isn't it?

Mathematical and Statistical Functions

The COBOL/370 Programming Guide has a good example of the power INTRINSIC FUNCTIONS offer in this area.

Suppose you have loaded a table with information about employees. Here's the table:

01 EMPLOYEE-TABLE.	
05 NUMBER-OF-EMPLOYEES	PIC S9(4) BINARY.
05 EMPLOYEE-RECORD	OCCURS 1 TO 500 TIMES
DEPENDING ON NUMBER-OF-EMPLOYEES.	
10 NAME	PIC X(20).
10 ID-NO	PIC 9(9).
10 SALARY	PIC 9(7)V99.

Now, would you like to know what your total expense for SALARY is? Here you go:

```
COMPUTE TOTAL-SALARIES          = FUNCTION SUM(SALARY(ALL))
```

Wow! Don't you love that index value of ALL? You can only use that in a FUNCTION, but it sure simplifies your code! Let's consider a few more questions:

What's the biggest SALARY paid?

```
COMPUTE MAXIMUM-SALARY          = FUNCTION MAX(SALARY(ALL))
```

(Continued on page 5)

COBOL/370 and LE/370, Article Four

(continued from page 4)

Who gets it? (As if we didn't know.)

COMPUTE INDEX-POINTER	= FUNCTION ORD-MAX(SALARY(ALL))
MOVE NAME(INDEX-POINTER)	TO HIGHEST-PAID-EMPLOYEE

What's the average SALARY?

COMPUTE AVERAGE-SALARY	= FUNCTION MEAN(SALARY(ALL))
------------------------	------------------------------

What's the difference between the maximum and minimum SALARY paid?

COMPUTE SALARY-DIFFERENTIAL	= FUNCTION RANGE(SALARY(ALL))
-----------------------------	-------------------------------

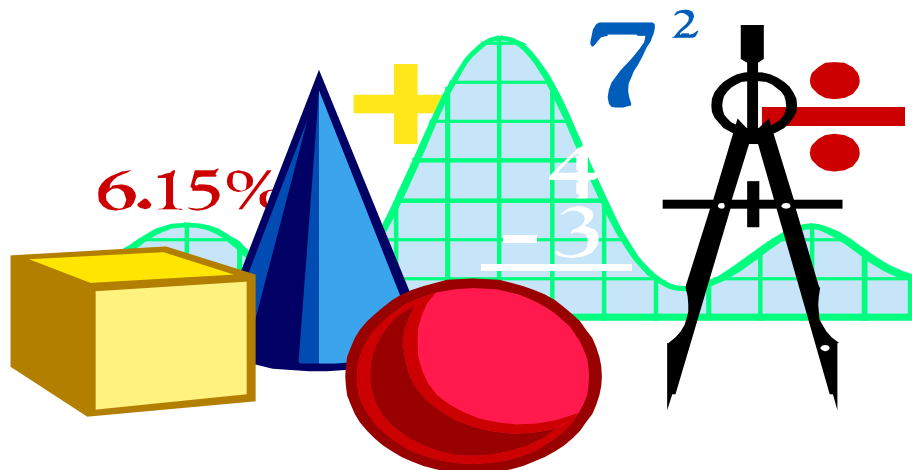
This completes our sampling of INTRINSIC FUNCTIONS. There are lots of other goodies in the category. Check them out!

More information about INTRINSIC FUNCTIONS and REFERENCE MODIFICATION can be found in:

IBM SAA AD/Cycle COBOL/370 Programming Guide (SC26-4767) and
IBM SAA AD/Cycle COBOL/370 Language Reference (SC26-4769)

The remaining articles of this series will be published in future issues of the *DIS Technical Broadcast*.

If you have any questions about these articles, please contact Gary Duffield at 902-3031. If you would like to obtain copies of all eight articles, contact Charie Martin at 902-3112. ☺



It's a Disaster!

 by Carol Criscione

The last weekend in October 1995 featured another Disaster Recovery test for Department of Information Services (DIS). I thought I'd share a glimpse of my perspective of the process in journal format. Enjoy!

October 4: Decisions are nearly complete about what to test and how. This is the annual "network test." The customers continue to work with the network staff coordinating which devices will be used in the test this time. Hopefully, most of the preparations for print testing are already complete. Who is scheduled to participate this time? DIS, HRISD, DSHS, ESD, DOL, OFM, L&I, DRS, OIC, DOT, and the Treasurer's Office.

October 11: The DIS Customer Disaster Recovery meetings continue. Next week is the last formal meeting before the test. Double-check my CICS documentation. Test CICS V4.1 in detail. We have three customer regions under it. Will also get a chance to test an IPL with the year 2000!

October 18: All should be fairly set for the test by now. Hopefully, all outstanding issues, projected test plans, etc., are complete and documented. My personal calendars for October 28-30 are 'clear.' The projected schedule shows most of my work should be on Saturday afternoon-Sunday night. The schedule shows many people will be testing from Olympia/Lacey.

October 23: Confirmed the hotel reservation in Seattle for Saturday night. Verified that the CICS related products do not require any special passwords to work at the disaster recovery hot site. Verified that the vendor phone numbers are included in the on-line documentation.

October 25: Ken Boling will be leaving for the site back East tonight. Verified the schedule of DIS staff departure dates and times. The test has begun. The tapes are on their way.

October 28: 9:50 P.M.: Went to the disaster recovery Seattle site and was briefed on the current status of the dataset restore process by Dave Smith (Disaster Recovery On-duty Coordinator.) I'll get to start testing CICS at approximately 2 A.M.

October 29: 01:14 A.M.: Phone rang but I couldn't find it due to the darkness and unfamiliarity with my environment. Waited for red message light so that I could find the phone.

01:45 A.M.: On site. Mark Foote (COA) starting all CICS regions. Will check interaction via logon screen for 40+ CICS regions.

02:53 A.M.: Can access all CICS regions. Mark resolved problem with reaching L&I regions via TPXLI. Continuing scripts for detailed testing.

(Continued on page 7)

It's A Disaster!

(Continued from page 6)

03:57 A.M.: Completed scripts. All looks GOOD for customer testing. Found a couple of minor problems and resolved them. The CICS testing portion of schedule was completed two hours ahead of projected time!

04:30 A.M.: ZZZZZZZ...

09:00 A.M.: Alarm went off. Please! Just 10 more minutes of sleep! Can I do without breakfast?

09:45 A.M.: Onsite for customer testing scheduled to begin at 10:00 A.M.

10:50 A.M.: Restored 3 datasets with Kim Starkey's assistance (DASD Support.) Will address problem at post-disaster recovery staff meetings.

12:05 P.M.: Reading; talking with customers and staff.

3:20 P.M.: More reading; talking with customers and staff. Feel like the Maytag repairperson...

3:35 P.M.: It's a nice day outside. I can see the ocean and wharf. There are some nicely restored buildings in the area, too. Noticed some people sitting outside, enjoying the sunshine, drinking various beverages, and talking. Looks interesting and I DO have a pager.

4:12 P.M.: Took a walk. The air smelled so 'autumny.' Didn't need a coat! Lots available downtown--people, events, SHOPPING! Talked with some 'locals.' Went to Starbucks; Bought a 'globe' mug full of double-shot coffee mocha.

4:55 P.M.: My hands stopped shaking from the coffeine.

5:15 P.M.: Working with Bob Hartsell (RACF Support) and Julia Bergevin (Adabas Support) to resolve problem with a DSHS transaction.

5:40 P.M.: Oh, oh! Another call for assistance!

5:50 P.M.: Allocated some ES files.

6:25 P.M.: One more problem to resolve. Collected documentation, two 'dumps,' three IOF listings for detailed analysis back at DIS.

7:50 P.M.: CICS regions being taken down. The test is 'winding down.' Post-customer data collection and dumping to cartridge starting. Collecting my notes for future documentation.

8:20 P.M.: CICS documentation collected for problem resolution put to cartridge by Don Schwarz (MVS Support) and Jim Collinsworth (DASD Support.) Man, am I tired!

(Continued on page 8)

It's A Disaster!

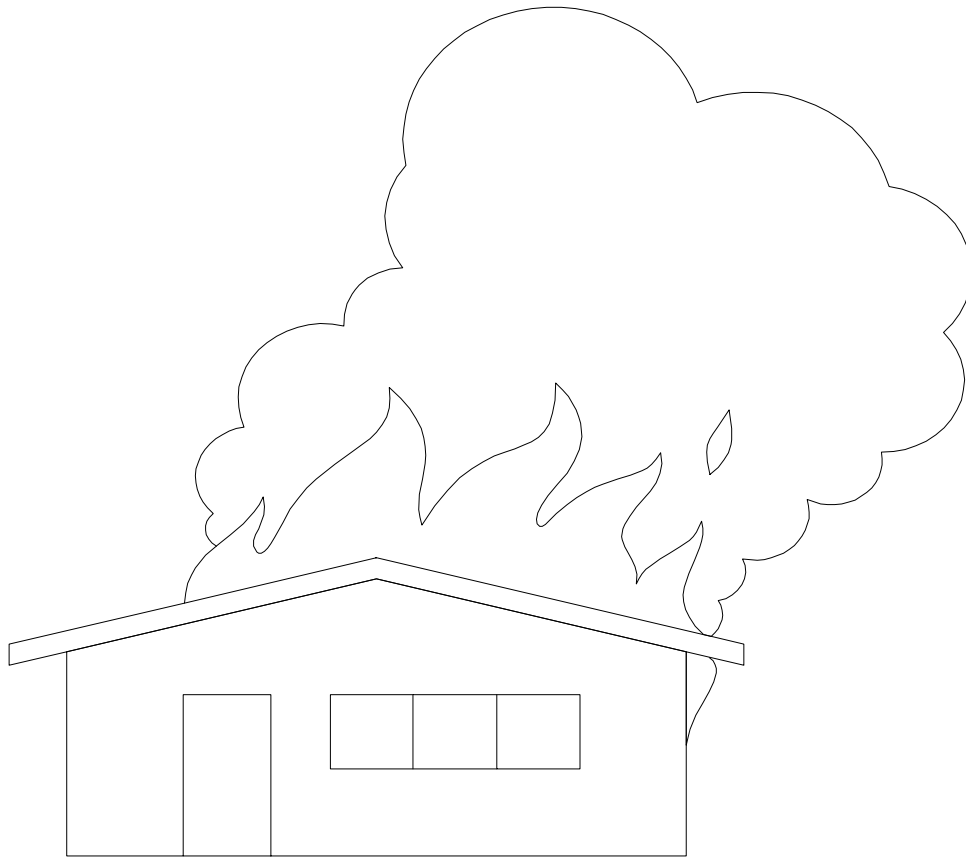
(Continued from page 7)

9:00 P.M.: IPLing V1 system with Year 2000 date. CICS test region started with no problem!

10:30 P.M.: On my way home. Unfortunately, I took the 'scenic route' to Bellevue prior to heading back to Olympia.

October 30: 12:05 A.M.: Home at last! My dog finally recognized me so I didn't get bitten. ZZZZZZZZZ.....

There it is--a snapshot of 'disaster recovering.' The next test is April 1996. Hopefully, your agency will be participating. It is a good chance to test recovery procedures and meet people. Until then! ☺



Making the Jump to Message Control Bank (MCB) by Steve Chiechi

Transaction Interface Processor (TIP), or lately, just Transaction Processing is the Online Transaction Processing system (OLTP) used on the Unisys 2200 mainframe. This is the first in a series of articles intended to aid programmers in writing, testing, and implementing transaction systems using TIP.

Since the installation of Unisys (then Sperry) mainframes at DIS in the mid-'70s, the transaction processing (TIP1100) environment has used a method of handling messages between user terminal and application program known as Communications Message POOL (COMPOOL). While COMPOOL is still supported, its use is being discouraged by Unisys and DIS in favor of a message handling system called Message Control Bank (MCB). COMPOOL support is not being enhanced, and will be discontinued in some future release of the 2200 Operating System. Meanwhile, new features and enhancements will support TIP utilizing MCB.

One note of caution regarding the order of calls to MCB and Data Management System (DMS) in an MCB Program: The DMS calls (IMPART/DEPART) must be totally contained within the TIP/MCB Initialization and Termination calls.

There are three basic methods for utilizing MCB:

Method 1 - MCB called via the Display Processing System (DPS 1100)

The preferred method for building a program utilizing MCB is to use DPS 1100. DPS provides a utility for interactive design and storage of user screens in a Form Library, and generation of working storage code for the application program. The calls to DPS in the program handle the interface to MCB.

This method is useful for new program development, and in those situations where a rewrite of the program replacing user-written screen code with DPS-defined screens and program logic is feasible.

Method 2 - MCB Application Program Interface

This method utilizes direct calls to MCB. The program will set up appropriate information in the MCB "packet" (a pre-defined data area provided via a COBOL COPY proc or

(Continued on page 10)

Making the Jump to Message Control Bank (MCB)

(Continued from page 9)

similar facility in other languages) then CALL the MCB to perform the requested function.

As described in Chapter 5 of the MCB Programming Reference Manual (PRM) functions include: Initialize, Read Input, Release Input, Store Output, Terminate, and others.

Method 3 - TIP Primitive Interface

For existing TIP programs currently using COMPOOL, an interface utilizing the same TIP primitive calls is available as a migration aid to MCB. These primitives offer a defined subset of MCB functionality as a compatible replacement of the functions available under COMPOOL.

In most cases, simply re-collecting (@MAP-ing) the program using the MCB library of TIP primitives and adding a “J” to the STATUS field and appropriate Application Group number in the AUD field of the program’s Validation Table (VALTAB) entry should work.

The following changes should be made to the program collection:

- At the beginning of the MAP directives, add a “NOT TIP\$*MCBLIB\$.CBEP\$\$MCB” statement
- On the LIB statement containing “TIP\$*TIPLIB\$”, change it to “TIP\$*MCBLIB\$”
- Add an “IN SY\$LIB\$*MCBn.CBEP\$\$MCBn” statement to the IBANK section of the MAP (where n=the APP Group number, 1 for DOL and 2 for DSHS)

The following two sample collections show the program “CSOLI” collected for COMPOOL, and then changed for MCB in APP Group 1.

(Continued on page 11)

Making the Jump to Message Control Bank (MCB)

(Continued from page 10)

```
-@MAP,L,NJR.CSOLI
-   MINGAP 65535
-   MINSIZ 65535
-   LIB (I1/$ODD,D1/$EVEN)
-   LIB SYS$LIB$*ACOB-TIPLIB(),SYS$LIB$*ACOB()
-   LIB SYS$LIB$*ACOB-TIPLIB(), TIP$*TIPLIB$()
-   .
-   .
-   .
-   IBANK,M I1
-   IN TPF$.CSOLI
-   DBANK,MC D1,070000
-   .
-   .
-   -END
```

```
-@MAP,S,NJR.CSOLI1
-   MINGAP 65535
-   MINSIZ 65535
-   NOT TIP$*MCBLIB$.CBEP$$MCB
-   LIB (I1/$ODD,D1/$EVEN)
-   LIB SYS$LIB$*ACOB-TIPLIB(),SYS$LIB$*ACOB()
-   LIB SYS$LIB$*ACOB-TIPLIB(), TIP$*MCBLIB$()
-   .
-   .
-   .
-   IBANK,M I1
-   IN SYS$LIB$*MCB1.CBEP$$MCB1
-   IN TPF$.CSOLI1
-   DBANK,MC D1,070000
-   .
-   -END
```

The following two screens show the VALTABs for the same programs, the first with no “J” in the STA field and zero AUD number, and the second with a “J” in the STA field and a “1” in the AUD field, for APP group 1 (DOL).

(Continued on page 12)

Making the Jump to Message Control Bank (MCB)

(Continued from page 11)

ATOVAL		VALTAB MAINTENANCE SYSTEM			
Program Number	82	ACT--Trans Code	CSOLI	NAM--Program Name:	CSOLI
PRG--Program Type: 1		OPT--Execution Options:		====IF MCB (STA = J)=====	
1 Self-destruct		LNRTZ		AUD--MCB audit number: 0	
RUN--Run time: 15		L TIP logging		QPR--Queue priority: 0	
		N Must sched output		REC--Recovery Options:	
		R Release compool		L MCB out msg audited	
		T DIAG\$ dump		R MCB inp msg audited	
COP--Prog Copies: 1		Z SEXEM dump		U Locks released	
PRT--Printer: TIPPR		STA--Prog status: _		X Rollback action	
		J=MCB, not compool		Y Commit action	
===== Static values are: =====					
LEV--Switch level 3		DBK--HVTIP size 0		IND--Load main storage	
MAS--Term class:		PCT--PCT size 2		PRI--Execute priority M	
		STF--Prog file 0		WAI--WAITQ length 5	
ACCOUNT: ACCT-BY-PID				ADD, DELETE, UPDATE OR RETURN: READ	
10/28/92 08:51:28 DCNAN					

ATOVAL		VALTAB MAINTENANCE SYSTEM			
Program Number	1823	ACT--Trans Code	CSOLI1	NAM--Program Name:	CSOLI1
PRG--Program Type: 1		OPT--Execution Options:		====IF MCB (STA = J)=====	
1 Self-destruct		LNRTZ		AUD--MCB audit number: 1	
RUN--Run time: 15		L TIP logging		QPR--Queue priority: 1	
		N Must sched output		REC--Recovery Options:	
		R Release compool		U	
		T DIAG\$ dump		L MCB out msg audited	
COP--Prog Copies: 1		Z SEXEM dump		R MCB inp msg audited	
PRT--Printer: TIPPR		STA--Prog status: J		U Locks released	
		J=MCB, not compool		X Rollback action	
===== Static values are: =====					
LEV--Switch level 3		DBK--HVTIP size 0		IND--Load main storage	
MAS--Term class:		PCT--PCT size 2		PRI--Execute priority M	
		STF--Prog file 0		WAI--WAITQ length 5	
ACCOUNT: ACCT-BY-PID				ADD, DELETE, UPDATE OR RETURN: READ	
10/23/92 08:42:59 DCNAN					

(Continued on page 13)

Making the Jump to Message Control Bank (MCB)


(Continued from page 12)

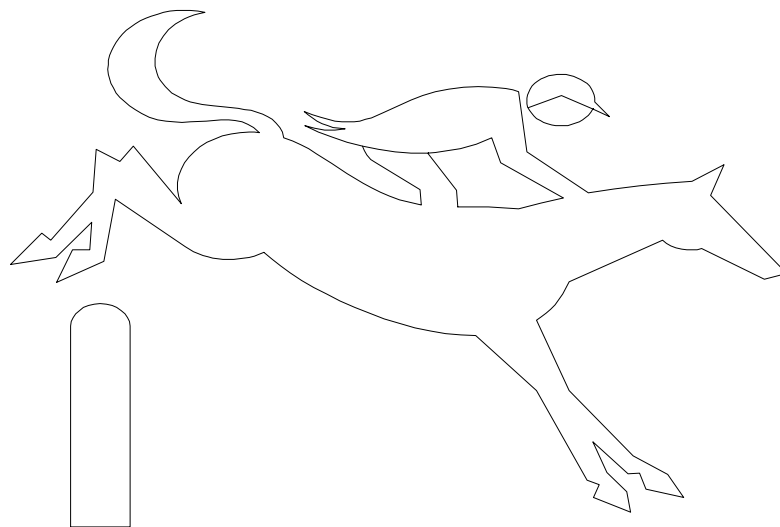
See Chapter 6 of the MCB PRM for considerations on each of the TIP primitives.

This method can be used to immediately and easily migrate existing TIP COMPOOL programs to MCB. Please note, however, that only a limited subset of MCB functions is supported by these primitives, and no enhancements are planned for them. In cases where the application programs will probably be around for a while, consideration should be given to investing in the extra work “now” of either method 1 or method 2, thus avoiding another conversion a few years down the road.

The following are Reference Manuals for OS1100 Message Control Bank Programming:

- MCB Programming Reference Manual (UP-13022)
- Display Processing System 1100 (DPS 1100) Run Time Programming Reference (7830 7840)
- Form Design Programming Guide (7831 2279) will provide more detail.

Please forward questions regarding this article or topics you’d like to see in future articles to Steve Chiechi at 902-3044, or send him an E-Mail: Steve.Chiechi. 



DIS Partners with the State Treasurer

 by Bonnie Beatty

The Office of the State Treasurer (OST) in partnership with the Department of Information Services (DIS), worked with all warrant producing agencies to redesign State warrant processing. The new warrant process went into effect on January 1, 1996.

Benefits include:

- Counterfeit warrants are reduced.
- Increased protection for banks and retailers, who cash state warrants from penalties over the original face value of the warrant.
- Reduced paper waste by splicing one warrant print job to the next.
- Reduced staff time and resources required to print warrants.
- Creation of an on-line system to eliminate hand-typed warrants.
- DIS created a common warrant print routine, so that future change to the face of the warrant would be transparent to warrant producing agencies.

Changes made by the OST to the new warrant document:

- **Watermarks** - A watermark has been added to the back side of the warrant and can be seen when the warrant is held at a 45 degree angle. This watermark does not reproduce on color copiers or computer scanners. A watermark has also been added below the pay amount area.
- **Toner Grip Paper** - State warrants are printed on “toner grip” paper that enhances the adhesion of the toner ink. This paper makes it difficult to remove the toner from the laser-printed warrant.
- **Secure Number Font** - The secure number font is a patented design that features a different size and shape for each number. The dollar amount is printed white on black,

(Continued on page 15)

DIS Partners with the State Treasurer

(Continued from page 14)

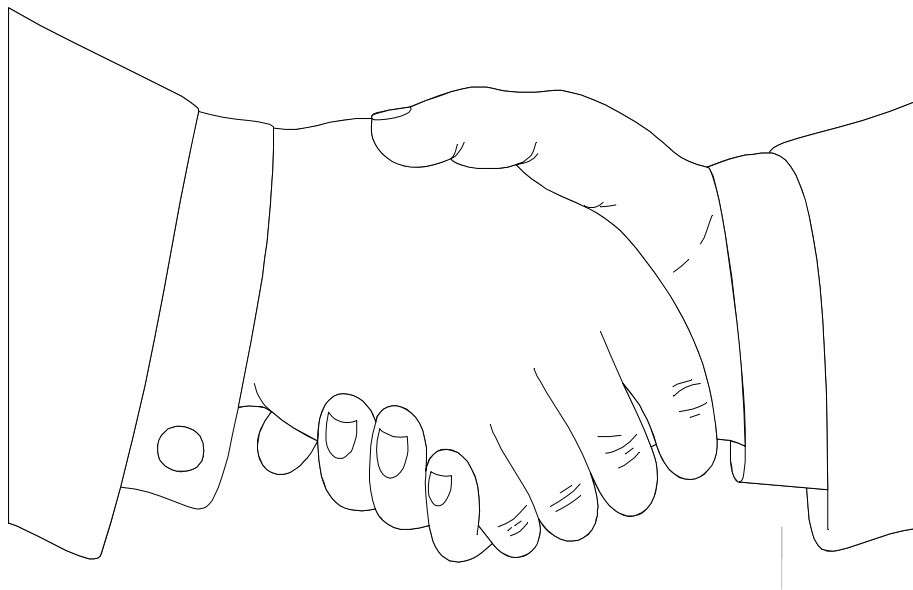
with the name of the number inside the number symbol. The cent amount is printed black on white, with the letters “CTS” running under each number. This unique font makes it virtually impossible to alter the warrant amount.

- **Dollar Amount Field** - The dollar amount in the pay amount field is printed in bold black numbers on a gray background as another deterrent to prevent alteration.
- **Agency Use Fields** - State agencies have the option of using one or both of these areas to print any information required by their agency, such as agency name, address, and phone number.

This is another demonstration of partnering with our customers to streamline state government. DIS is dedicated to supporting its customers to meet their information technology challenges and business requirements.

NOTE: SEE SAMPLE WARRANT ON THE NEXT PAGE.

If you have any questions about the new warrant process, please contact Gordon Bowman at 902-3205. If you have any questions about the warrant document, please contact Gayleen Cox, Office of the State Treasurer at 902-8901. 📞



The Washington State Treasurer has re-designed state warrants (checks) with new security features. The new design includes several technological advances that will make it more difficult to counterfeit or alter state warrants. All state warrants have been updated to include these new features.

Toner Grip Paper: State warrants are printed on 'toner grip' paper which enhances the adhesion of the toner ink. This paper makes it difficult to remove the toner from the laser-printed warrant.

Agency Use Areas: State agencies have the option of using one or both of these areas to print any information required by their agency, such as agency name, address, and phone number.

Watermark: A watermark has been added to the **back side** of the warrant and can be seen when the warrant is held a 45 degree angle. This watermark does not reproduce on color copiers or computer scanners. A watermark has also been added below the pay amount

WASHINGTON STATE WATERMARK APPEARS IN PAY AMOUNT FIELD

STATE OF WASHINGTON
OFFICE OF STATE TREASURER OLYMPIA
D.S.H.S. AGENCY 300
PHONE 360 902-9999
98-557 000791C
1251

DEPT OF SOCIAL & HEALTH SERV.
P. O. BOX 888
OLYMPIA WA 98504

Reg No.	Agency	Sub. Ag.	Warrant No.	Mo.	Day	Yr.
P4344	3000	00	000791C	01	01	1995

PAY TO THE ORDER OF
JANE DOE
1234 MAIN STREET
ANYTOWN WA 99999

PAY ONLY **2139 56** CTS
NEGOTIABLE FOR 180 DAYS OR AFTER ABOVE DATE
DANIEL K GRIMM, STATE TREASURER

PAY THIS AMOUNT
\$2,139.56

⑈00079103⑈ ⑆125105576⑆ 09007900⑈

Account Number: An account number has been added to the MICR line which allows state warrants to validate through check verification systems. Counterfeit warrants that do not have magnetic ink in the MICR line can be detected immediately on the check verification readers used in many retail stores.

Secure Number Font: The secure number font is a patented design that features a different size and shape for each number. The dollar amount is printed white on black, with the name of the number inside the number symbol. The cent amount is printed black on white, with the letters "CTS" running under each number. This unique font makes it virtually impossible to alter the amount line.

Dollar Amount Field: The dollar amount in the pay amount field is printed in bold black numbers on a gray background as another deterrent to prevent alteration.

Why Restore an Archived Data Set just to Delete It? by Kim Starkey

When you need to delete disk data sets, the tendency is to use ISPF option 3.4 Data Set List Utility. If the data sets you are deleting are archived, an auto restore is initiated. The auto restore process submits a job, and a tape is mounted to restore the disk data set.

By accessing SAMS: Disk DMS/OS Panels you will be able to delete a disk data sets without initiating the restore process or waiting for it to expire.

From the DIS Main Menu, select option **S**, and from the DIS Storage Utilities Menu, select **DMS**. This will take you to DMS/OS Selection Menu, and from here select **2** for LIST - (OR DELETE) ARCHIVE/BACKUP INDEX ENTRIES. In the Data Set Name field, enter the fully qualified data set name or a masked data set name, and in the Version field, enter **A** to indicate all copies.

EXAMPLE:

```
DMS010L ----- Deffered Request Command Processing -- SAMS:DISK --
COMMAND ==> L

ISPF Library:
PROJECT ==>
LIBRARY ==>
TYPE ==>

Press END key to terminate processing.
Press ENTER to process request.

Other data set, cluster name, or SAMS:DISK pattern name for select list:

DATA SET NAME ==> 'TEST155./'

Request type ==> a      Enter blank or R for restore, A for archive.
```

At this point, the archived and backup copies will be listed if any exist. Enter a **D** next to the entries you want to delete from the list.

EXAMPLE:

(Continued on page 18)

Why Restore an Archived Data Set just to Delete It?

(Continued from page 17)

DMS007 ----- SAMS:Disk Archived Data Set Index Selection ROW 1 TO 8 OF 8
COMMAND INPUT ===> SCROLL ===> CSR

Archived	- Archived -
Data Set Name	Typ. Date Time Org.
-----	--- -----
TEST155.DATASET.ONE	BKP 06JAN1996 08.25 PO
D TEST155.DATASET.TWO	BKP 23DEC1995 07.09 PO
TEST155.DATASET.THREE	ARC 12JAN1996 19.37 PS

There will be a confirmation screen before deleting or canceling the request.

EXAMPLE:

DMS009H ----- Confirm Delete ----- SAMS:DISK
COMMAND ===> L

Data set name:	EXCH155.DATASET.TWO
Original volser:	VSYS04
Backup date:	23DEC1995
Backup time:	07:09
Backup expiration date:	25JAN1996
Archvols key:	116375

Press ENTER to confirm delete request.

Press END key to cancel delete request.

(Continued on page 19)

Why Restore an Archived Data Set just to Delete It?

(Continued from page 18)

Four types of MASKING are available for this process:

1. **CHARACTER MASKING** - The question mark (?) may be used to represent any single character at that location in node or simple name. Multiple occurrences may be used within each node level or simple name.

EXAMPLE:

====> ? will select all single-character data set names
====> A.TEST?? will select all two-level data set names with a
 high level qualifier, and a simple name of six characters,
 the first four of which must be TEST. For example, TEST01,
 TEST02, TESTLA, etc.

2. **LEVEL MASKING** - The asterisk (*) may be used to represent any node level or simple name.

EXAMPLE:

====> * will select all single-level data set names
====> *.* will select all two-level data set names
====> A.*.LIB will select all three-level names that have A as the high level
 qualifier and LIB as the simple name, but any second-level node.
====> *.*.JCL will select all data sets that have JCL as the third level node.

NOTE: Use caution when masking the high level qualifier as it may cause DMS/OS to process the entire control file and incur excessive TSO CPU charges to process such requests.

3. **ANYTHING AFTER MASKING** - The slash (/) may be used to represent any character(s) from that position to the end of the name. The portion of the name that precedes the slash is referred to as a prefix name.

EXAMPLE:

(Continued on page 20)

Why Restore an Archived Data Set just to Delete It?

(Continued from page 19)

- ====> A/ will select all data sets that begin with letter A
- ====> A.TEST/ will select all data sets that begin with the character string 'A.TEST' with any character following.
- ====> A.*.C?./ will select all data sets that begin with a high level qualifier of A, followed by any second level node, a two-character third level node starting with letter C, and any following string.

The appropriate use of the slash is to mask anything after a point in a data set name.

- 4. ANYTHING BEFORE/AFTER MASKING** - The exclamation point (!) may be used to represent any character(s) at the beginning of a data set name. The exclamation point is used to request a match for the characters following the exclamation point anywhere within the data set name.

EXAMPLE:

- ====> !TEST will select all data sets that contain test somewhere in the name.
- ====> A?.!DEPT2 will select all data set that have a two-character first level index starting with A and contains DEPT2 somewhere in the name.
- ====> !TEST!LIB will select all data sets that contain TEST somewhere in the name and LIB somewhere following TEST.

NOTE: Use extreme caution when coding exclamation point. We highly recommend narrowing the scope of the mask by using other patterns mask or specific coding qualifiers prior to the exclamation point.

A data set name that is not enclosed in quotes will be prefixed with the user's TSO USERID, and data set name enclosed in quotes will not be prefixed.

EXAMPLE:

- ====> XYZ DSN=USERID.XYZ
- ====> / DSN=USERID./
- ====> 'XYZ' DSN=XYZ
- ====> '/' DSN=/'

We hope you find this informative and that it will helps you save storage costs and processing time! ☺

the Technical Broadcast



Charie L. Martin, Editor
Department of Information Services
Adams Building
1310 Jefferson ST SE
Mail Stop: 42445
Olympia, WA 98504-2445

Technical Broadcast Staff - The *Technical Broadcast* is published quarterly by the Department of Information Services (DIS). The purpose is to provide a forum for customer information-sharing of upcoming system enhancements and optimization tips. We invite your articles, comments, and suggestions.

DIS is an equal opportunity employer and does not discriminate on the basis of race, religion, color, sex, age, nationality, or disability.



Washington State Department of
Information Services
